Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L8	10	(Jan near2 gray).in.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:37
L9	2096	quer$6 with cache	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:37
L10	623064	(determin$4 or estimat$5) with time	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:37
L11	827	L9 and L10	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:37
L12	120506	multilevel or hierarch$5	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:37
L13	415	L11 and L12	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:37
L14	1519883	semiconductor	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:37
L15	16	L13 and L14	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:37
L16	25565	"711"/$.ccls.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:37

L17	25565	"711"/$.ccls.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:37
L18	48	L13 and L17	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:38
L19	6774	(cache near hit)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 11:53
L20	382152	instruction cache	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:38
L21	8510	instruction adj cache	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:39
L22	18129	data adj cache	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:39
L23	3787	21 with 22	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:39
L24	94	19 same 23	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:40
L25	3143	cache near2 way$2	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 11:54
L26	26	24 and 25	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:43

L27	3438	cache near2 (spa$4 or localit$2)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:49
L28	311	cache near2 (localit$2)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:55
L29	57	cache near2 (spatial)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:55
L30	65811	referenc$4 near2 time	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 10:55
L31	8	29 and 30	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 11:51
L32	1734	cache near3 residen$4	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 11:52
L33	4062	"load/store"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 11:52
L34	108	32 and 33	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 11:53
L35	36	23 and 34	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 11:53
L36	8094	(cache near2 hit)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 11:53

L37	15	35 and 36	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 11:53
L38	9572	cache near2 (way$2 or level$2 or hierarch$4)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 11:55
L39	11	37 and 38	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 11:55
L40	600855	(software or (operating adj system))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 11:56
L41	8	39 and 40	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/07/18 11:56

Terms used
hierarchical cache instruction data cache cache hit software operating system

Found **404** of **157,873**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown

Relevance scale

1 Session 3B: Compiler techniques in system level design: Software-assisted cache replacement mechanisms for embedded systems

Prabhat Jain, Srinivas Devadas, Daniel Engels, Larry Rudolph

November 2001 **Proceedings of the 2001 IEEE/ACM international conference on Computer-aided design**

Full text available: pdf(193.73 KB) Additional Information: full citation, abstract, references, citings, index terms

We address the problem of improving cache predictability and performance in embedded systems through the use of software-assisted replacement mechanisms. These mechanisms require additional software controlled state information that affects the cache replacement decision. Software instructions allow a program to kill a particular cache element, i.e., effectively make the element the least recently used element, or keep that cache element, i.e., the element will never be evicted.We prove basic th ...

2 Multi-level shared caching techniques for scalability in VMP-M/C

D. R. Cheriton, H. A. Goosen, P. D. Boyle

April 1989 **ACM SIGARCH Computer Architecture News , Proceedings of the 16th annual international symposium on Computer architecture**, Volume 17 Issue 3

Full text available: pdf(1.27 MB) Additional Information: full citation, abstract, references, citings, index terms

The problem of building a scalable shared memory multiprocessor can be reduced to that of building a scalable memory hierarchy, assuming interprocessor communication is handled by the memory system. In this paper, we describe the VMP-MC design, a distributed parallel multi-computer based on the VMP multiprocessor design, that is intended to provide a set of building blocks for configuring machines from one to several thousand processors. VMP-MC uses a memory hierarchy based on shared caches ...

3 Research sessions: non-standard query processing: Buffering databse operations for enhanced instruction cache performance

Jingren Zhou, Kenneth A. Ross

June 2004 **Proceedings of the 2004 ACM SIGMOD international conference on Management of data**

Full text available: pdf(188.52 KB) Additional Information: full citation, abstract, references

As more and more query processing work can be done in main memory access is becoming a significant cost component of database operations. Recent database research has shown

that most of the memory stalls are due to second-level cache data misses and first-level instruction cache misses. While a lot of research has focused on reducing the data cache misses, relatively little research has been done on improving the instruction cache performance of database systems.We first answer the question "Why ...

4 Characterizing the caching and synchronization performance of a multiprocessor operating system

Josep Torrellas, Anoop Gupta, John Hennessy

September 1992 **ACM SIGPLAN Notices , Proceedings of the fifth international conference on Architectural support for programming languages and operating systems**, Volume 27 Issue 9

Full text available: pdf(1.52 MB) Additional Information: full citation, references, citings, index terms

5 System-level power optimization: techniques and tools

Luca Benini, Giovanni de Micheli

April 2000 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 5 Issue 2

Full text available: pdf(385.22 KB) Additional Information: full citation, abstract, references, citings, index terms

This tutorial surveys design methods for energy-efficient system-level design. We consider electronic sytems consisting of a hardware platform and software layers. We consider the three major constituents of hardware that consume energy, namely computation, communication, and storage units, and we review methods of reducing their energy consumption. We also study models for analyzing the energy cost of software, and methods for energy-efficient software design and compilation. This survery ...

6 Application-specific memory management for embedded systems using software-controlled caches

Derek Chiou, Prabhat Jain, Larry Rudolph, Srinivas Devadas

June 2000 **Proceedings of the 37th conference on Design automation**

Full text available: pdf(76.30 KB) Additional Information: full citation, abstract, references, citings, index terms

We propose a way to improve the performance of embedded processors running data-intensive applications by allowing software to allocate on-chip memory on an application-specific basis. On-chip memory in the form of cache can be made to act like scratch-pad memory via a novel hardware mechanism, which we call column caching. Column caching enables dynamic cache partitioning in software, by mapping data regions to a specified sets of cache "columns" or "ways ...

7 Data and memory optimization techniques for embedded systems

P. R. Panda, F. Catthoor, N. D. Dutt, K. Danckaert, E. Brockmeyer, C. Kulkarni, A. Vandercappelle, P. G. Kjeldsberg

April 2001 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 6 Issue 2

Full text available: pdf(339.91 KB) Additional Information: full citation, abstract, references, citings, index terms

We present a survey of the state-of-the-art techniques used in performing data and memory-related optimizations in embedded systems. The optimizations are targeted directly or indirectly at the memory subsystem, and impact one or more out of three important cost metrics: area, performance, and power dissipation of the resulting implementation. We first examine architecture-independent optimizations in the form of code transoformations. We next cover a broad spectrum of optimizati ...

Keywords: DRAM, SRAM, address generation, allocation, architecture exploration, code transformation, data cache, data optimization, high-level synthesis, memory architecture customization, memory power dissipation, register file, size estimation, survey

8 Exploiting cache affinity in software cache coherence

Hui Li, Kenneth C. Sevcik

July 1994 **Proceedings of the 8th international conference on Supercomputing**

Full text available: pdf(999.44 KB) Additional Information: full citation, abstract, references, citings, index terms

Cache affinity is important to the performance of scalable shared memory multiprocessors. For multiprocessors without hardware cache coherence support, software cache coherence is the only alternative. Most existing software cache schemes ignore cache affinity across parallel loops. In this paper, we propose a new scheme, Cache Affinity-based Software cache coherence scheme (CAS), that exploits cache affinity across parallel loops to achieve high cache hit ratios without requiring extra har ...

9 Cache coherence in large-scale shared-memory multiprocessors: issues and comparisons

David J. Lilja

September 1993 **ACM Computing Surveys (CSUR)**, Volume 25 Issue 3

Full text available: pdf(3.12 MB) Additional Information: full citation, references, citings, index terms

10 Data cache locking for higher program predictability

Xavier Vera, Björn Lisper, Jingling Xue

June 2003 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 2003 ACM SIGMETRICS international conference on Measurement and modeling of computer systems**, Volume 31 Issue 1

Full text available: pdf(292.01 KB) Additional Information: full citation, abstract, references, citings, index terms

Caches have become increasingly important with the widening gap between main memory and processor speeds. However, they are a source of unpredictability due to their characteristics, resulting in programs behaving in a different way than expected.Cache locking mechanisms adapt caches to the needs of real-time systems. Locking the cache is a solution that trades performance for predictability: at a cost of generally lower performance, the time of accessing the memory becomes predictable.This pape ...

Keywords: data cache analysis, worst-case execution time

11 A version control approach to Cache coherence

Hoichi Cheong, Alex Veidenbaum

June 1986 **Proceedings of the 3rd international conference on Supercomputing**

Full text available: pdf(1.03 MB) Additional Information: full citation, abstract, references, citings, index terms

A version control approach to maintain cache coherence is proposed for large-scale shared-memory multiprocessor systems with interconnection networks. The new approach, unlike existing approaches for such class of systems, makes it possible to exploit temporal locality across synchronization boundaries. As with the other software-directed approaches, each processor independently manages its cache, i.e., there is no interprocessor communication involved in maintaining cache coherence ...

Keywords: parallel task execution, software-directed cache coherence, version control

12 Comparative performance evaluation of cache-coherent NUMA and COMA architectures

Per Stenström, Truman Joe, Anoop Gupta

April 1992 **ACM SIGARCH Computer Architecture News , Proceedings of the 19th annual international symposium on Computer architecture**, Volume 20 Issue 2

Full text available: pdf(1.52 MB) Additional Information: full citation, abstract, references, citings, index terms

Two interesting variations of large-scale shared-memory machines that have recently emerged are cache-coherent non-uniform-memory-access machines (CC-NUMA) and cache-only memory architectures (COMA). They both have distributed main memory and use directory-based cache coherence. Unlike CC-NUMA, however, COMA machines automatically migrate and replicate data at the main-memory level in cache-line sized chunks. This paper compares the performance of these two classes ...

13 A memory management unit and cache controller for the MARS system

Feipei Lai, Chyuan-Yow Wu, Tai-Ming Parng

November 1990 **Proceedings of the 23rd annual workshop and symposium on Microprogramming and microarchitecture**

Full text available: pdf(1.07 MB) Additional Information: full citation, abstract, references

For large caches, the interaction between cache access and address translation affects the machine cycle time and the access time to memory. The physically addressed caches slow down the cache access due to the virtual address translation. The virtually addressed caches is faster, but the synonym problem is difficult to handle. By some software constraints and hardware support, our virtually addressed physically tagged caches can achieve the same speed as traditional virtually addressed cac ...

14 Informing memory operations: memory performance feedback mechanisms and their applications

Mark Horowitz, Margaret Martonosi, Todd C. Mowry, Michael D. Smith

May 1998 **ACM Transactions on Computer Systems (TOCS)**, Volume 16 Issue 2

Full text available: pdf(344.74 KB) Additional Information: full citation, abstract, references, citings, index terms, review

Memory latency is an important bottleneck in system performance that cannot be adequately solved by hardware alone. Several promising software techniques have been shown to address this problem successfully in specific situations. However, the generality of these software approaches has been limited because current architecturtes do not provide a fine-grained, low-overhead mechanism for observing and reacting to memory behavior directly. To fill this need, this article proposes a new class ...

Keywords: cache miss notification, memory latency, processor architecture

15 Two-level hierarchical register file organization for VLIW processors

Javier Zalamea, Josep Llosa, Eduard Ayguadé, Mateo Valero

December 2000 **Proceedings of the 33rd annual ACM/IEEE international symposium on Microarchitecture**

Full text available: pdf(154.90 KB) ps(843.85 KB) Publisher Site Additional Information: full citation, references, citings, index terms

16 Special system-oriented section: the best of SIGMOD '94: AlphaSort: a cache-sensitive parallel external sort

Chris Nyberg, Tom Barclay, Zarka Cvetanovic, Jim Gray, Dave Lomet

October 1995 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 4 Issue 4

Full text available: pdf(1.37 MB) Additional Information: full citation, abstract, references, citings

A new sort algorithm, called AlphaSort, demonstrates that commodity processors and disks can handle commercial batch workloads. Using commodity processors, memory, and arrays of SCSI disks, AlphaSort runs the industry-standard sort benchmark in seven seconds. This beats the best published record on a 32-CPU 32-disk Hypercube by 8:1. On another benchmark, AlphaSort sorted more than a gigabyte in one minute. AlphaSort is a cache-sensitive, memory-intensive sort algorithm. We argue that modern arch ...

Keywords: Alpha, Dec 7000, cache, disk, memory, parallel, sort, striping

17 Analysis of multiprocessor cache organizations with alternative main memory update policies

W. C. Yen, K. S. Fu

May 1981 **Proceedings of the 8th annual symposium on Computer Architecture**

Full text available: pdf(1.04 MB) Additional Information: full citation, abstract, references, citings, index terms

Cache memory has played a significant role in the memory hierarchy and has been used extensively in large systems and minisystems. The effectiveness of cache memories with alternative main memory update policies in a multiprocessor system is a major concern in this paper. The performances of write-through with write-allocation or no-write allocation, buffered write-through, flag-swap, and buffered flag-swap policies have been analyzed. Because of the dominating cost of the interface between ...

18 Multithreading I: Pointer cache assisted prefetching

Jamison Collins, Suleyman Sair, Brad Calder, Dean M. Tullsen

November 2002 **Proceedings of the 35th annual ACM/IEEE international symposium on Microarchitecture**

Full text available: pdf(1.21 MB) Publisher Site Additional Information: full citation, abstract, references, citings, index terms

Data prefetching effectively reduces the negative effects of long load latencies on the performance of modern processors. Hardware prefetchers employ hardware structures to predict future memory addresses based on previous patterns. Thread-based prefetchers use portions of the actual program code to determine future load addresses for prefetching.This paper proposes the use of a pointer cache, which tracks pointer transitions, to aid prefetching. The pointer cache provides, for a given pointer's ...

19 Transactional client-server cache consistency: alternatives and performance

Michael J. Franklin, Michael J. Carey, Miron Livny

September 1997 **ACM Transactions on Database Systems (TODS)**, Volume 22 Issue 3

Full text available: pdf(452.41 KB) Additional Information: full citation, abstract, references, citings, index terms, review

Client-server database systems based on a data shipping model can exploit client memory resources by caching copies of data items across transaction boundaries. Caching reduces the need to obtain data from servers or other sites on the network. In order to ensure that

such caching does not result in the violation of transaction semantics, a transactional cache consistency maintenance algorithm is required. Many such algorithms have been proposed in the literature and, as all provide the sam ...

20 Informing memory operations: providing memory performance feedback in modern processors

Mark Horowitz, Margaret Martonosi, Todd C. Mowry, Michael D. Smith

May 1996 **ACM SIGARCH Computer Architecture News , Proceedings of the 23rd annual international symposium on Computer architecture**, Volume 24 Issue 2

Full text available: pdf(1.55 MB) Additional Information: full citation, abstract, references, citings, index terms

Memory latency is an important bottleneck in system performance that cannot be adequately solved by hardware alone. Several promising software techniques have been shown to address this problem successfully in specific situations. However, the generality of these software approaches has been limited because current architectures do not provide a fine-grained, low-overhead mechanism for observing and reacting to memory behavior directly. To fill this need, we propose a new class of memory operati ...